

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 08-044498

(43)Date of publication of application : 16.02.1996

(51)Int. Cl. G06F 3/06

G06F 12/02

G06F 12/04

(21)Application number : 06-177986

(71)Applicant : HITACHI LTD

(22)Date of filing : 29.07.1994

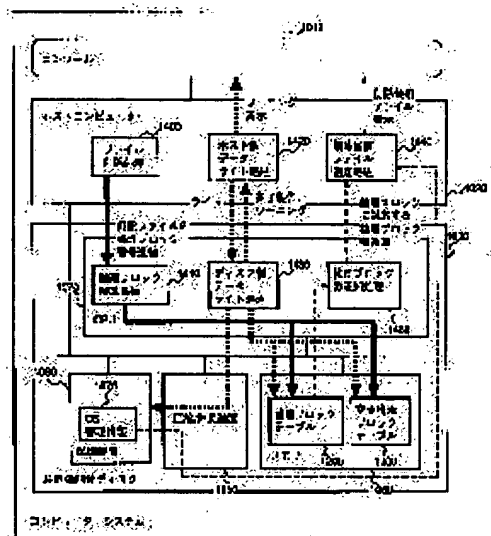
(72)Inventor : ACHIWA KIYOSUKE
YAMAMOTO AKIRA
FUJII TETSUHIKO
YAMAGATA HIROTAKE
KOBASHI TETSUZO

(54) STORAGE DEVICE WITH COMPRESSING FUNCTION AND COMPUTER SYSTEM HAVING THE DEVICE

(57)Abstract:

PURPOSE: To previously prevent a physical application area from being filled up by informing a host computer of warning at the time of judging the approach of a filled state in the physical application area and allowing the host computer to output the warning to a console.

CONSTITUTION: In each deletion of a file by the host computer 1020 or the change of a logical block to be allocated to a file by data writing processing, a logical block whose data storage is made unnecessary is informed to a compressing function-added disk 1030 so as not to store unnecessary data, so that the application efficiency of a disk 1030 can be improved. The reduction of free physical blocks is informed to a user by displaying a warning message 'Free areas in the disk are reduced.' on a console 101. When a message 'Delete an unnecessary file.' is displayed, a following processing method is suggested to the user and the user deletes a file judged as an unnecessary one.



BEST AVAILABLE COPY

1 of 2

LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision
of rejection][Kind of final disposal of application
other than the examiner's decision of
rejection or application converted
registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's
decision of rejection][Date of requesting appeal against
examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998, 2003 Japan Patent Office

JP08-044498

*** NOTICES ***

JPO and NCIP are not responsible for any damages caused by the use of this translation.

1. This document has been translated by computer. So the translation may not reflect the original precisely.
2. **** shows the word which can not be translated.
3. In the drawings, any words are not translated.

CLAIMS

[Claim(s)]

[Claim 1] The store with a compression function connected to the calculating machine is the store with a compression function characterized by to have a means output warning to said calculating machine when a means compress the data from said calculating machine which consists of two or more logical blocks, the means which assigns said compressed data to the physical block which is the physical data-storage unit of said store, and said physical block which should be assigned decrease.

[Claim 2] In the computer system which consists of storage with a compression function connected with the computer which has an output unit at it said storage with a compression function A means to compress the data from said calculating machine which consists of two or more logical blocks, The means which assigns said compressed data to the physical block which is the physical data storage unit of said store, And it is the storage with a compression function which has a means to output warning to said computer when said physical block which should be assigned decreases, and is characterized by said computer having a means to output said warning from said storage with a compression function to said output unit.

[Claim 3] The computer system characterized by having a means to display the utilization situation of a physical data storage field in a computer system according to claim 2 in case said warning is outputted to said output unit.

[Claim 4] The computer system characterized by having a means to display the size of the part which does not store data among said physical blocks on said output unit in a computer system according to claim 3.

[Claim 5] The computer system characterized by having a means to display the size of the part which stores data among said physical blocks on said output unit in a computer system according to claim 3.

[Claim 6] The computer system which is a computer system which has the storage with a compression function which shows the bigger logical address than a actual physical data storage field as a host computer, and a host computer, and has the means which writes in data with sufficient compressibility to the logical address which the host computer does not use.

[Claim 7] The computer system which is a computer system which has the store with a compression function which shows the bigger logical address than a actual physical data storage field as a host computer, and a host computer, and has a means to judge the logical address which is not used based on the management domain of an operating system, and the means which writes in data with sufficient compressibility to the logical address which the host computer does not use.

[Claim 8] The store with a compression function which provides a host computer with a bigger logical data storage field than a actual physical data storage field by compressing data, and has a means output the number of physical blocks which is a store with a compression function with the function which assigns the logical block which is the access unit of a host computer to the physical block which is the physical data storage unit of a store, and supports the logical block according to the demand of a host computer to a host computer.

[Claim 9] A host computer is provided with a bigger logical data storage field than a actual physical data storage field by compressing data. It has the function which assigns the logical block which is the access unit of a host computer to the physical block which is the physical data storage unit of a store. The storage with a compression function which has a means to output the number of physical blocks corresponding to a logical block to a host computer according to the demand of a host computer, The computer system which is a computer system which has a host computer and has a means to change the number of physical blocks corresponding to a logical block into the number of physical blocks corresponding to a file, based on the management domain of an operating system.

DETAILED DESCRIPTION

[Detailed Description of the Invention]

[0001]

[Industrial Application] This invention relates to the storage which is applied to the storage which has a compression function, especially notifies the difference of a actual storing capacity and a logical capacity to a host computer.

[0002]

[Description of the Prior Art] In a computer system, since disk capacity is used effectively, the compression function of data may be given to a disk unit. PCT/US 91/04270 The technique of improving the utilization effectiveness in a disk with a compression function is indicated. With this technique, whenever a user publishes a file deletion demand, it notifies to a disc system, and the field where the file set as the object of deletion was stored is made into an usable field.

[0003] In the usual OS, in order not to notify deletion of a file to a disk only by rewriting the management information inside OS in case a file is deleted, a disk continues holding unnecessary data until the light to the address next comes. For this reason, it is the conventional technique when there is compression, although it does not become a problem especially when there is no compression. PCT/US 91/04270 A technique as shown is needed. That is, although the disk with a compression function shows the logical address more than a actual physical capacity as the host computer A physical capacity actual before carrying out the light of the data corresponding to all the logical addresses depending on the compressibility of the data to store is used up (a physical activity field is full). Overwrite of the light of the data corresponding to the logical address after it and data with compressibility lower than the data stored from the first becomes impossible. Also in order to avoid such a situation if possible, it should be made not to have unnecessary data as much as possible.

[0004]

[Problem(s) to be Solved by the Invention] There are the following technical problems in the above-mentioned conventional technique. the above-mentioned conventional technique PCT/US 91/04270 **** -- since file management which was conscious of a physical capacity was not performed, the physical activity field filled and there was a problem of stopping being able to

carry out the light of the new data.

[0005] the above-mentioned conventional technique PCT/USs 91/04270 **** -- since transfer of the frequent information for publishing the command with which a host computer tells a disk about deletion of a file, and a disk recognizing it, and releasing a disk field, in order to make available the field on the disk which the eliminated file had occurred, there was a problem that the overheads of file deletion increased in number. Moreover, in order to realize the above-mentioned function, there was a problem that large modification of a program was required.

[0006] The 1st object of this invention is offering the storage with a compression function which can prevent in advance a physical activity field filling. The 2nd object of this invention is offering the implementation approach which reduces the overhead of the file deletion which makes available the field on the disk which the eliminated file's had, and lessens the amount of modification of the program for it.

[0007]

[Means for Solving the Problem] In order to attain the 1st object of the above, the following processings are performed in this invention. First, in a store with a compression function, there is no corresponding logical block, and the number of physical blocks which can assign a new logical block decreases, when it judges that the physical activity field approached to the limit, warning is notified to a host computer and a host computer outputs warning to a console. Then, a user can know that saw warning and the uncommitted storage list field with a compression function approached to the limit, and the available field on a disk can be increased by deleting some unnecessary files. However, since a user does not know the size of the physical field on the disk which each file occupies, he cannot increase the efficient available field on a disk. Then, when a user is told about a physical area size on the disk which each file occupies by ***** shown below and a user determines the file which looks at and deletes it, the efficient available field on a disk can be increased. That is, the storage with a compression function notifies a physical area size on the storage corresponding to a logical management unit to a host computer, and a host computer displays a file name on a console at descending of the physical field of the storage which the logical management unit and logical file on storage are made to correspond, and supports the file.

[0008] In order to attain the 2nd object of the above, a host computer judges the field where the file on a store is not assigned based on the management information of an operating system, and carries out the light of the data with sufficient compressibility there.

[0009]

[Function] According to this invention, the store with a compression function does not have a corresponding logical block, and the number of physical blocks which can assign a new logical block decreases, when judged with the physical activity field having approached to the limit, warning is reported to a host computer and a host computer outputs it to a console.

Consequently, a user can see warning outputted to the console, it can know that the physical activity field is approaching to the limit, the free area of a disk with a compression function can be increased by deleting an unnecessary file, and it can prevent a physical activity field filling by this as much as possible.

[0010] The storage with a compression function notifies a physical area size corresponding to the logical management unit of storage to a host computer according to the demand from a host computer. Based on the management information of OS, a host computer makes the management unit and file of logic of a store correspond, and gets to know a physical area size which each file occupies on a disk. And the console output of the file name is carried out at descending of the

physical field occupied on a disk. Consequently, in case a user deletes a file, he can specify the file which can increase a free area efficiently as an object for deletion with reference to this console output.

[0011] A host computer judges the field to which a file is not assigned on the store based on the management information of an operating system, and performs data light processing for carrying out the light of the data with sufficient compressibility there. Data light processing of a host computer requests data light processing from a store with a compression function to write data with sufficient compressibility in the address demanded from the store with a compression function. Data light processing of a store with a compression function compresses data with sufficient compressibility with a light demand, and carries out a light to the demanded address. And the field on the disk currently assigned to the demanded address from the first is released, and a free area is secured. At this time, the field which stores the small compressed data which compressed data with sufficient compressibility in the address specified by the light demand instead of the field on the disk currently assigned from the first is secured, and when it is many, the free area on a disk will increase. The above-mentioned processing is not performed synchronizing with the usual file deletion, but when a user starts an application program, it carries out. For this reason, an overhead excessive to the usual deletion is not applied, and modification of OS does not have the need, either.

[0012]

[Example]

The 1st example drawing 1 is the computer system 1000 which applied this invention. First, a configuration is explained. A computer system 1000 consists of consoles 1010 for compressing data and transmitting the information from the memorizable disk 1030 with a compression function (storage), a host computer 1020, and a host computer 1020 to a user. Furthermore, the disk 1030 with a compression function consists of storage 1090 which stores the compression expanding equipment 1050 which performs compression processing and expanding processing of data, CPU1070, memory 1060, and compressed data 1140.

[0013] Drawing 2 shows the relation between the logical disk 1100 which appears from a host computer 1020, and the actual physical store 1090, and matching of a logical block and a physical block in the disk 1030 with a compression function.

[0014] In this example, capacity of 2GB and storage is set to 1GB for the capacity of a logical disk 1100 in consideration of the compression effectiveness. The logical disk 1100 consists of logical blocks 1120 with the size of 4KB, and the store 1090 consists of physical blocks 1110 with the size of 512B. A host computer 1020 carries out access to the disk 1030 with a compression function to logical-block 1120 unit.

[0015] Next, signs that the data of the logical block a1121 on a logical disk 1100 are actually stored in storage 1090 are shown in drawing 2.

[0016] The expanding data a1131 which are data of a logical block a1121 are compressed, and turn into compressed data a1141, and division storing is carried out at physical BUROKU a1112, a physical block b1113, and a physical block c1114. Generally, since the size of compressed data 1141 is not fixed, the number of the physical blocks 1112-4 required to store it becomes one - eight pieces. However, compressed data 1141 does not become larger than the expanding data 1131.

[0017] The table which has actually managed matching of a logical block 1121 and a physical block 1112-4 is the logic block table 1200. The logic block table entry 1211-3 for several logical-block minutes is shown in the logic block table 1200, and there is an entry of the physical

block number 1290 to the 1230-8th physical block numbers of the 1220 or 1st compressed data size among each logic block table entries 1211-3. The size of the compressed data 1141 when compressing the expanding data 1131 of the logical block 1121 is stored in the compressed data size 1220. Moreover, the number of the physical blocks 1112-4 corresponding to a logical block 1121 can be obtained from this compressed data size 1220. The a maximum of eight physical block number which stores compressed data 1141 is stored in the physical block number 1290 of the 1st physical block numbers [1230-8th]. When the number of the physical blocks 1112-4 corresponding to a logical block 1121 is seven or less, the value which has not been made into the physical block number, -1 [for example,], is stored in the entry in which the physical block 1112-4 which corresponds among the physical block numbers 1290 of the 1st physical block numbers [1230-8th] does not exist. Moreover, it enables it to distinguish from the case of others by storing the compression size 0 and storing a value -1 in the entry of the physical block number 1290 of the 1st physical block numbers [1230-8th] to the logical block 1121 in which the corresponding physical block 1112-4 does not exist.

[0018] Drawing 3 is an empty physics block table which does not store effective data and which is vacant and manages a physical block 1112-4. It consists of empty physical block bit maps 1320 which indicate [an opening or] it whether that is right to be 1310 empty physical blocks by 1 bit to all the physical blocks 1112-4 on a store 1090 to the empty physics block table 1300, respectively. Are vacant, and 1 is made to correspond in this example while using 0.

[0019] Next, the outline of this invention is described using drawing 1. In a computer system 1000, the number of the logical block 1121 corresponding to the file which the file deletion 1400 of a host computer 1020 deleted to the logical-block release processing 1410 by CPU1070 on the occasion of deletion of a file is notified, the logical-block release processing 1410 is vacant with the logic block table 1200 on memory 1060, and the data of the file which rewrote and deleted the physical block table 1300 are made not to be held. The arrow head of a thick line shows the above-mentioned procedure.

[0020] The arrow head of a thick dotted line shows processing of the data light in a computer system 1000. First, if the data light processing 1420 of a host computer 1020 carries out the light of the data to the disk 1030 with a compression function, and the data light processing 1430 of CPU1070 can store data in a store 1090, light data will be compressed with compression expanding equipment 1050, and it will write in a store 1090, and will be vacant with the logic block table 1200 on memory 1060, the physical block table 1300 will be corrected, and light completion will be reported to a host computer 1020. And if there is a logical block 1121 which newly became an opening, the data light processing 1420 by the side of a host notifies the number of the logical block 1121 to the disk 1030 with a compression function, it will be vacant as for the data light processing 1430 by the side of a disk with the logic block table 1200, will correct the physical block table 1300, and will release the physical block 1112-4 corresponding to the logical block 1121 used as an opening. On the other hand, if data are unstorable in a store 1090, the data light processing 1430 by the side of a disk reports warning to a host computer 1020, and the data light processing 1420 by the side of a host displays warning on a console 1010.

[0021] The arrow head of a thin broken line shows the processing which outputs a deletion candidate's file name and capacity to a console. First, the advice processing of a physical block corresponding to a logical block of CPU1070 investigates the number of physical blocks corresponding to each logical block 1121 based on the logic block table 1200 stored in memory 1060, and notifies it to a host computer 1020. And based on the OS management information

1470, the deletion candidate file selection processing 1440 of a host computer 1020 makes a file and a logical block 1121-3 correspond, makes the response result and the information sent from the disk 1030 with a compression function associate, and investigates how many physical blocks 1112-4 each file supports, respectively. And a file name is sorted in many [of the corresponding more than physical block 1112-4] order, and it outputs to a console 1010.

[0022] The above-mentioned procedure is explained to a detail using a flow chart. The file deletion in a computer system 1000 is explained using drawing 4 and drawing 5. Drawing 4 is the flow chart of the file deletion 1400 in a host computer 1020. Processing 1400 adds the function for treating a disk with a compression function to the file deletion of OS. First, the file as which deletion was required is deleted at step 2000. At step 2010, a logical-block release command is published to a disk 1030, and the number of the logical block 1121-3 to which the eliminated file was assigned is notified.

[0023] Drawing 5 is the flow chart of the logical-block release processing 1410 in a disk with a compression function. First, at step 2100, the number of a logical block 1121-3 is received from a host computer 1020. At step 2110, it is vacant with the logic block table 1200, the physical block table 1300 is rewritten, and the physical block 1112-4 corresponding to the logical block 1120 which received the number is made into an opening.

[0024] Next, the data light processing in a computer system 1000 is explained using drawing 6, drawing 7, and drawing 8. Drawing 6 is the flow chart of the data light processing 1430 of the disk 1030 with a compression function. First, at step 2200, from a host computer 1020, light data are received and it stores in memory 1060. Since this light data is in an incompressible condition, it is called the expanding data 1131. Next, at step 2210, compression expanding equipment 1050 compresses the expanding data 1131 on memory 1060, generates compressed data 1141, and stores it in memory 1060. And CPU1070 looks for a number required to store compressed data 1141 of empty physical blocks 1110 using the empty physical block bit map 1320 (step 2220). The light of the compressed data 1140 on memory 1060 is carried out to the empty physical block 1110 found at step 2220 (step 2230).

[0025] CPU1070 rewrites the logic block table entry 1211-3 corresponding to the logical block 1121 with a light demand, and makes a logical block 1121 and a physical block 1112-4 correspond at step 2240. Furthermore, at step 2250, CPU1070 subtracts the number of physical blocks which was stored in the empty physics block table 1300 and which it was vacant and carried out the light from 1310 physical blocks, and sets 0 which shows under an activity as the bit corresponding to a physical block [finishing / the light in the empty physical block bit map 1320]. Moreover, the physical block 1112-4 which supported the logical block 1121 as which the light was required from the first is released as an opening, the number is added to 1310 empty physical blocks, and 1 is set as the bit to which the empty physical block bit map 1320 corresponds.

[0026] At step 2251, it judges whether there is an empty physical block 1112-4 of the capacity in which CPU1070 can store light data based on 1310 empty physical blocks of the empty physics block table 1300. And if blocks 1112-4 are insufficient, it will jump to step 2252. If blocks 1112-4 are insufficient, at step 2260, CPU1070 judges whether the empty physical block 1110 fully remains based on 1310 empty physical blocks of the empty physics block table 1300. In this example, if there is 10% or more, it will consider as the thing more than [whose] are all the physical blocks more than 1110 in the empty physical block 1110 and it is judged that fully remains.

[0027] And if judged with the number of the empty physical blocks 1110 fully remaining, it will

jump to step 2280. Although CPU1070 performs a light completion report to a host computer 1020 at step 2270 when judged with the number of the empty physical blocks 1110 fully not remaining, the empty physical block 1110 reports warning which shows few things at this time. At step 2290, it is judged whether the logical-block release request is coming from the host computer 1020. If the logical-block release request is not coming, the light processing in the disk 1030 with a compression function is ended. If the logical-block release request is coming, at step 2300, logical-block release processing 1410 shown in drawing 5 will be performed, and the light processing in the disk 1030 with a compression function will be ended.

[0028] On the other hand, at step 2280, although CPU1070 performs a light completion report to a host computer 1020, at this time, it does not perform a warning report, but reports normal termination, and jumps it to step 2290.

[0029] On the other hand, at step 2252, it reports that it is vacant to a host computer 1020, the number of physical blocks 1110 is insufficient, and light data cannot be stored in a disk, and the data light processing in the disk 1030 with a compression function is ended.

[0030] Drawing 7 is the flow chart of the data light processing 1420 of a host computer 1020. First, at step 2400, a light demand is published to the disk 1030 with a compression function. At step 2410, light data are sent to the disk 1030 with a compression function. At step 2420, it will be in the condition of the waiting for completion of light processing of the disk 1030 with a compression function. At step 2430, the completion report from the disk 1030 with a compression function is received. If it is judged whether warning which shows that there are few empty physical blocks 1110 from the disk 1030 with a compression function at step 2440 was reported and there is no warning, it will jump to step 2460. When warning is reported, at step 2450, warning is outputted to a console 1010 and a solution is suggested to a user. At step 2460, it is judged whether the logical block which newly became an opening exists. The above-mentioned processing is produced when the logical block 1120 which OS assigns to a file is changed by overwrite of a file etc. If the logical block 1120 which newly became an opening does not exist, data light processing of a host computer 1020 is ended. If the empty logical block 1120 exists, at step 2470, the release request of a logical block 1120 will be published by the disk 1030 with a compression function, the number of a logical block 1120 used as an opening will be transmitted, and data light processing of a host computer 1020 will be ended.

[0031] The utilization effectiveness of the disk 1030 with a compression function can be raised by the logical block 1120 whose need for data-hold was lost on the disk 1030 with a compression function as stated above whenever it deleted the file with the host computer 1020, or whenever the logical block 1120 currently assigned to the file was changed by data light processing being notified, and unnecessary data being made not to be held.

[0032] Drawing 8 is the example of warning displayed on a console 1010. By the display of "the free area of a disk has decreased", the empty physical block 1110 has notified the user of few things. In the display of "please delete an unnecessary file", the future solution is suggested to the user. By deleting the file judged to be unnecessary, a user is an empty physical block in a physical block. 1110 A number can be increased.

[0033]

"It is the amount of object hairdressing during an activity. : The amount of XXXXKB usable object hairdressing : The amount of YYYYKB total object hairdressing : ZZZZKB"

A user can know the utilization situation of storage 1090 from *****. By this display, a user can know the rule of thumb of the amount of cutbacks of a file.

[0034] And a user deletes an unnecessary file, when the amount of usable object hairdressing is

not enough, some can be copied to other disks, tapes, etc., a file can be deleted from the disk 1030 with a compression function, and **** of a physical activity field can still be prevented by increasing the empty physical block 1110.

[0035] Finally, drawing 9 and drawing 13 are used and the processing which shows a user a deletion candidate's file is explained. In case some files which a user deletes are chosen, a big difference will appear in the number of the empty physical blocks 1110 which newly becomes usable in the time of only the good file of compressibility being chosen, and the time of only the bad file of compressibility being chosen. Therefore, in this example, when a file is eliminated, a file name is outputted to order with many physical blocks corresponding to a console 1010 so that a user can choose a file to which as many physical blocks 1110 as possible serve as an opening as an object for deletion.

[0036] Drawing 9 is the flow chart of the advice processing 1450 of the number of physical blocks of the disk 1030 with a compression function corresponding to a logical block. First, at step 2600, the number of physical blocks 1110 corresponding to each logical block 1120 is investigated based on the logic block table 1200. At step 2610, the number of the physical blocks 1110 corresponding to each logical block 1120 is sent to a host computer 1020.

[0037] Drawing 13 is the flow chart of the deletion candidate file selection processing 1440 of a host computer. First, at step 2700, a demand which notifies the number of physical blocks 1110 which supports the disk 1030 with a compression function at each logical block 1120 is published. At step 2710, the data of the more than physical block 1110 corresponding to each logical block 1120 are received from the disk 1030 with a compression function. At step 2720, as shown in drawing 10, based on the OS management information 1470, a logical block 1120 and a file are associated and the number of the physical blocks 1110 corresponding to each file is investigated. For example, when OS of a host computer 1020 is MS-DOS, based on FAT (File Allocation Table), the response relation between a file and a logical block 1120 can be known. At step 2730, a file name is outputted to many [of the more than physical block 1110 corresponding to the last] sequence at a console 1010.

[0038] Drawing 11 is the example of a screen output of an elimination candidate file. The information which consists of a file name, after [compression] size, and size before compression is displayed in order of the file size after compression for every file. The size after compression of a file will be obtained if physical block 1110 size is applied [more than] in the corresponding physical block 1110. Moreover, the size before compression of a file is obtained from the OS management information 1470. A user can see a console, can know which file occupies many disks 1030 with a compression function, and chooses the file which should delete this as one of the selection ingredients.

[0039] In the 2nd example [1st] of an example, the physical block 1110 corresponding to the logical block 1120 which notified directions of deletion to the disk whenever it deleted the file, in order to solve the problem of the utilization effectiveness of a disk with a compression function, and was assigned to the eliminated file was released. However, by the above-mentioned approach, since it performed synchronizing with the deletion and light processing of a file, there was a fault that the overhead accompanying the synchronization of processing became large. Moreover, the processing for notifying the lost logical block 1120 to a disk, and making the corresponding physical block 1110 into an opening needed to be made to OS, and the modification magnitude was large.

[0040] Drawing 12 is the outline of invention corresponding to this example. In order to reduce the overhead of elimination processing of a file, or light processing and to make modification

magnitude of OS small in this example, when it is not notified at every deletion of a file at a disk but the usable empty physical block 1110 has decreased, it is collecting into all the logical blocks 1120 without the need for data-hold, and writing in data with sufficient compressibility, and the free-area recovery application 1460 which increases more than in the empty physical block 1110 is formed. By processing 1460, more than is increased in the usable empty physical block 1110, and the problem of the utilization effectiveness of the disk 1030 with a compression function is coped with.

[0041] The arrow head of a thick dotted line shows data light processing of a computer system 1000. First, if the data light processing A1421 of a host computer 1020 carries out the light of the data to the disk 1030 with a compression function, if the data light processing A1431 of CPU1070 can store data on a store 1090, it compresses light data with compression expanding equipment 1050, writes the result in a store 1090, will be vacant with the logic block table 1200 of memory 1060, will correct the physical block table 1300, and will report light completion to a host computer 1020. On the other hand, if data storage cannot be carried out to a store 1090, the disk side data light processing A1431 reports warning to a host computer 1020, and the host side data light processing A1421 displays warning on a console 1010.

[0042] The arrow head of a thin broken line shows the processing which carries out the console output of a deletion candidate's file. First, based on the logic block table 1200 in memory 1060, the processing which notifies the physical block corresponding to the logical block of CPU1070 investigates the number of physical blocks corresponding to each logical block 1120, and notifies it to a host computer 1020. The deletion candidate file selection processing 1440 of a host computer 1020 makes a file and a logical block 1120 correspond based on the OS management information 1470, this and the sent information are made to associate, and each file investigates how many physical blocks 1110 are supported, respectively. A file name is sorted by many [of the corresponding more than physical block 1110] order, and it is outputted to a console 1010 at it.

[0043] The arrow head of a thin continuous line shows processing of the free-area recovery application 1460 of a host computer 1020. First, the logical block 1120 which OS is not using is judged based on the OS management information 1470, and a light demand is given to the host side data light processing A1421 so that the data of oar 0 may be written in all the intact logical blocks 1120.

[0044] Drawing 14 is deleted in a file, and before and after writing data with sufficient compressibility in the logical block 1120 from which storing data became unnecessary, it shows the example whose empty physical block 1110 of the disk 1030 with a compression function seems to increase.

[0045] The expanding data m1135 are stored in the logical block k1125 before elimination of a file. The expanding data m1135 are compressed, turn into compressed data m1145, and are stored in six physical blocks 1110. When the good expanding data n1136 of compressibility are written in after file deletion, the expanding data n1136 are compressed, and turn into compressed data n1146, and compressed data n1146 is stored in one physical block 1110. At this time, the five remaining physical blocks 1110 turn into the empty physical block 1110.

[0046] Hereafter, a different point from the 1st example is explained. Logical-block release processing 1410 shown at every deletion of a file by this example at drawing 5 since the logical block 1120 corresponding to the deleted file was not notified to a disk, and step of drawing 4 2010 Since there is no need, it becomes like drawing 15 .

[0047] Moreover, since the logical block 1120 whose need for data-hold was lost also at the time

of a file light is not notified to a disk, step 2290 of drawing 6 , step 2300, and step 2460 and step 2470 of drawing 7 do not have the need. Drawing 6 and drawing 7 become like drawing 16 and drawing 17 , respectively.

[0048] The warning screen corresponding to this example which carries out a console output becomes like drawing 18 . The display of "please run free-area recovery application" is added for the following reasons. Only by deleting a file in this example, since the usable physical block 1110 in a disk with a compression function cannot be made to increase, it is because it is necessary to release the physical block 1110 in which the close data of the file which was made to perform free-area recovery application 1460, and was deleted are.

[0049] Drawing 19 is the flow chart of the free-area recovery application 1460. At step 2500, the logical block 1120 which is not used is judged based on the management domain of OS of a host computer 1020. For example, if OS of a host computer 1020 is MS-DOS, based on FAT (File Allocation Table), the above-mentioned decision is possible. When OS has the management information to which a file and a free block are managed also in other OS's, the logical block 1120 which is not used can be judged with reference to this management information. At step 2510, the data of oar 0 are written in the logical block 1120 judged to be intact. Here, the data of oar 0 are shown as an example of data with sufficient compressibility, and as long as they are data with comparatively sufficient compressibility, what kind of data are sufficient as them. However, the data with which compressed data 1140 is restored to one physical block 1110 are desirable. At this example, it shall become the size restored to one physical block by compressing the data of oar 0.

[0050] As mentioned above, by performing free-area recovery application 1460, in the disk 1030 with a compression function, the physical block 1110 which is continuing having data of the deleted file can be released, and the empty physical block 1110 can be increased.

[0051]

[Effect of the Invention] According to this invention, the store with a compression function does not have a corresponding logical block, and the number of physical blocks which can assign a new logical block decreases, when judged with the physical activity field having approached to the limit, warning is reported to a host computer and a host computer outputs it to a console. Consequently, a user can see warning outputted to the console, it can know that the physical activity field is approaching to the limit, the free area of a disk with a compression function can be increased by deleting an unnecessary file, and it can prevent a physical activity field filling by this as much as possible.

[0052] The storage with a compression function notifies a physical area size corresponding to the logical management unit of storage to a host computer according to the demand from a host computer. Based on the management information of OS, a host computer makes the management unit and file of logic of a store correspond, and gets to know a physical area size which each file occupies on a disk. And the console output of the file name is carried out at descending of the physical field occupied on a disk. Consequently, in case a user deletes a file, he can specify the file which can increase a free area efficiently as an object for deletion with reference to this console output.

[0053] A host computer judges the field to which a file is not assigned on the store based on the management information of an operating system, and performs data light processing for carrying out the light of the data with sufficient compressibility there. Data light processing of a host computer requests data light processing from a store with a compression function to write data with sufficient compressibility in the address demanded from the store with a compression

function. Data light processing of a store with a compression function compresses data with sufficient compressibility with a light demand, and carries out a light to the demanded address. And the field on the disk currently assigned to the demanded address from the first is released, and a free area is secured. At this time, the field which stores the small compressed data which compressed data with sufficient compressibility in the address specified by the light demand instead of the field on the disk currently assigned from the first is secured, and when it is many, the free area on a disk will increase. The above-mentioned processing is not performed synchronizing with the usual file deletion, but when a user starts an application program, it carries out. For this reason, an overhead excessive to the usual deletion is not applied, and modification of OS does not have the need, either.

[Translation done.]

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平8-44498

(43) 公開日 平成8年(1996)2月16日

| (51) Int.Cl. ⁶ | 識別記号 | 庁内整理番号 | F I | 技術表示箇所 |
|---------------------------|---------|---------|-----|--------|
| G 0 6 F 3/06 | 3 0 1 W | | | |
| 12/02 | 5 3 0 D | 7623-5B | | |
| 12/04 | 5 3 0 | 7623-5B | | |

審査請求 未請求 請求項の数 9 O L (全 15 頁)

(21) 出願番号 特願平6-177986

(22) 出願日 平成6年(1994)7月29日

(71) 出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72) 発明者 阿知和 恭介

神奈川県川崎市麻生区王禅寺1099番地 株式会社日立製作所システム開発研究所内

(72) 発明者 山本 彰

神奈川県川崎市麻生区王禅寺1099番地 株式会社日立製作所システム開発研究所内

(72) 発明者 藤井 哲彦

神奈川県川崎市麻生区王禅寺1099番地 株式会社日立製作所システム開発研究所内

(74) 代理人 弁理士 小川 勝男

最終頁に続く

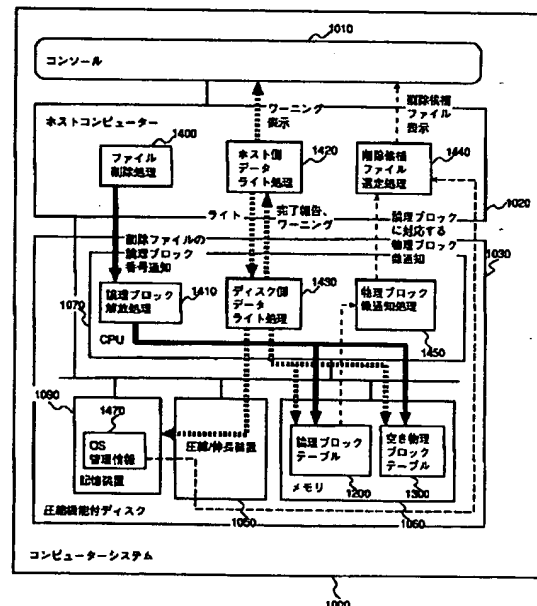
(54) 【発明の名称】 圧縮機能付き記憶装置及びそれを有するコンピュータシステム

(57) 【要約】

【目的】 実際の物理的な容量よりも、大きな論理アドレスをホストコンピュータに見せる圧縮機能付き記憶装置において、物理的な記憶領域が全て使用され、満杯になることを防ぐ。

【構成】 圧縮機能付きディスク1030は空き物理ブロック1110数がある程度以下になったら、ホストコンピュータ1020にワーニングを報告し、ホストコンピュータ1020はコンソール1010にワーニングを出力してユーザーに満杯が近いことを知らせ、ユーザーにファイルの削除を依頼することで、圧縮機能付きディスク1030が満杯になることを防ぐ。

図1



1

【特許請求の範囲】

【請求項1】 計算機に接続された圧縮機能付記憶装置は、

複数の論理ブロックからなる前記計算機からのデータを圧縮する手段、

前記圧縮されたデータを前記記憶装置の物理的なデータ格納単位である物理ブロックに割り当てる手段、及び割り当てるべき前記物理ブロックが少なくなった時に、前記計算機にワーニングを出力する手段を有することを特徴とする圧縮機能付記憶装置。

【請求項2】 出力装置を有する計算機とそれに接続された圧縮機能付記憶装置とからなるコンピュータシステムにおいて、前記圧縮機能付記憶装置は、

複数の論理ブロックからなる前記計算機からのデータを圧縮する手段、

前記圧縮されたデータを前記記憶装置の物理的なデータ格納単位である物理ブロックに割り当てる手段、及び割り当てるべき前記物理ブロックが少なくなった時に、前記計算機にワーニングを出力する手段を有し、

前記計算機は、前記圧縮機能付記憶装置からの前記ワーニングを前記出力装置に出力する手段を有することを特徴とする圧縮機能付記憶装置。

【請求項3】 請求項2記載のコンピュータシステムにおいて、前記出力装置に前記ワーニングを出力する際に、物理的なデータ格納領域の利用状況を表示する手段を有することを特徴とするコンピュータシステム。

【請求項4】 請求項3記載のコンピュータシステムにおいて、前記物理ブロックのうちデータを格納していない部分のサイズを前記出力装置に表示する手段を有することを特徴とするコンピュータシステム。

【請求項5】 請求項3記載のコンピュータシステムにおいて、前記物理ブロックのうちデータを格納している部分のサイズを前記出力装置に表示する手段を有することを特徴とするコンピュータシステム。

【請求項6】 実際の物理的なデータ格納領域よりも大きな論理アドレスをホストコンピュータに見せる圧縮機能付記憶装置とホストコンピュータを有するコンピュータシステムであって、ホストコンピュータが利用していない論理アドレスに対して、圧縮率の良いデータを書き込む手段を有するコンピュータシステム。

【請求項7】 実際の物理的なデータ格納領域よりも大きな論理アドレスをホストコンピュータに見せる圧縮機能付記憶装置とホストコンピュータを有するコンピュータシステムであって、オペレーティングシステムの管理領域に基づいて、利用されていない論理アドレスを判定する手段と、ホストコンピュータが利用していない論理アドレスに対して、圧縮率の良いデータを書き込む手段を有するコンピュータシステム。

【請求項8】 データを圧縮することによって、実際の物理的なデータ格納領域よりも大きな論理的データ格納領域

2

をホストコンピュータに提供し、ホストコンピュータのアクセス単位である論理ブロックを記憶装置の物理的なデータ格納単位である物理ブロックに割り当てる機能を持つ圧縮機能付記憶装置であって、ホストコンピュータの要求に応じて、論理ブロックに対応している物理ブロック数をホストコンピュータに出力する手段を有する圧縮機能付記憶装置。

【請求項9】 データを圧縮することによって、実際の物理的なデータ格納領域よりも大きな論理的データ格納領域をホストコンピュータに提供し、ホストコンピュータのアクセス単位である論理ブロックを記憶装置の物理的なデータ格納単位である物理ブロックに割り当てる機能を持ち、ホストコンピュータの要求に応じて、論理ブロックに対応している物理ブロック数をホストコンピュータに出力する手段を有する圧縮機能付記憶装置と、ホストコンピュータを有するコンピュータシステムであって、オペレーティングシステムの管理領域に基づいて、論理ブロックに対応する物理ブロック数を、ファイルに対応する物理ブロック数に変換する手段を有するコンピュータシステム。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明は、圧縮機能を有する記憶装置に係り、特に実際の格納容量と論理的な容量との差をホストコンピュータに通知する記憶装置に関する。

【0002】

【従来の技術】 コンピュータシステムでは、ディスク容量を有効に使うため、ディスク装置にデータの圧縮機能を持たせることがある。PCT/US91/04270 は、圧縮機能を持つディスクにおける利用効率を改善する技術を開示している。この技術では、ユーザーがファイル削除要求を発行する度に、ディスクシステムに通知し、削除の対象となるファイルが格納されていた領域を使用可能な領域にする、というものである。

【0003】 通常のOSにおいて、ファイルを削除する際にOS内部の管理情報を書き換えるだけで、ディスクにはファイルの削除を通知しないために、ディスクは次にそのアドレスへのライトが来るまで、不要なデータを保持し続ける。このため、圧縮が無い場合は特に問題にはならないが、圧縮がある場合には、従来技術 PCT/US91/04270 に示すような技術が必要となる。つまり、圧縮機能を持つディスクは、実際の物理的な容量以上の論理アドレスをホストコンピュータに見せているが、格納するデータの圧縮率によっては、全ての論理アドレスに対応するデータをライトする前に実際の物理的な容量を使い切ってしまう（物理的な使用領域が満杯）、それ以降の論理アドレスに対応するデータのライトや、元々格納されていたデータよりも圧縮率の低いデータの上書きができなくなることがある。このような状況をできるだけ避けるためにも、不要なデータは極力持たないようにす

べきである。

【0004】

【発明が解決しようとする課題】上記従来技術には、以下のような課題がある。上記従来技術 PCT/US91/04270 では、物理的な容量を意識したファイル管理が行なわれていないので、物理的な使用領域が満杯になって、新たなデータがライトできなくなるという問題があった。

【0005】上記従来技術 PCT/US91/04270 では、削除したファイルが持っていたディスク上の領域を利用可能とするために、ホストコンピュータがファイルの削除をディスクに知らせるコマンドを発行し、ディスクがそれを認識してディスク領域を解放するための頻繁な情報の授受が発生するため、ファイル削除処理のオーバーヘッドが増えるという問題があった。また、上記の機能を実現するために、プログラムの大幅な改造が必要であるという問題があった。

【0006】本発明の第1の目的は、物理的な使用領域が満杯になることを事前に防止できる、圧縮機能付記憶装置を提供することである。本発明の第2の目的は、削除したファイルが持っていたディスク上の領域を利用可能にするファイル削除処理のオーバーヘッドを低減し、またそのためのプログラムの改造量を少なくする実現方法を提供することである。

【0007】

【課題を解決するための手段】上記第1の目的を達成するために、本発明では、以下の処理を行う。まず、圧縮機能付記憶装置において、対応する論理ブロックがなく、かつ、新たな論理ブロックの割り当てが可能な物理ブロック数が少なくなって、物理的な使用領域が満杯に近付いたと判定されたとき、ホストコンピュータにワーニングを通知し、ホストコンピュータは、コンソールにワーニングを出力する。その後、ユーザはワーニングを見て圧縮機能付記憶装置の使用可能領域が満杯に近付いたことを知り、不要なファイルをいくつか削除することで、ディスク上の利用可能な領域を増やすことができる。ただし、ユーザは各ファイルが占有しているディスク上の物理的な領域のサイズを知らないため、効率良くディスク上の利用可能な領域を増やすことができない。そこで、次に示す手順により、各ファイルが占有しているディスク上の物理的な領域サイズをユーザに知らせ、ユーザはそれを見て削除するファイルを決定することにより、効率良くディスク上の利用可能な領域を増やすことができる。すなわち、圧縮機能付記憶装置は、論理的な管理単位に対応する記憶装置上の物理的な領域の大きさをホストコンピュータに通知し、ホストコンピュータは、記憶装置上の論理的な管理単位とファイルに対応させ、ファイルに対応している記憶装置の物理的な領域の大きい順に、ファイル名をコンソールに表示する。

【0008】上記第2の目的を達成するために、ホストコンピュータは、オペレーティングシステムの管理情報

に基づいて、記憶装置上のファイルの割り当てられていない領域を判定し、そこへ圧縮率の良いデータをライトする。

【0009】

【作用】本発明によれば、圧縮機能付記憶装置は、対応する論理ブロックがなく、かつ、新たな論理ブロックの割り当てが可能な物理ブロック数が少なくなって、物理的な使用領域が満杯に近づいたと判定されたとき、ホストコンピュータにワーニングを報告し、ホストコンピュータはそれをコンソールに出力する。その結果、ユーザはコンソールに出力されたワーニングを見て、物理的な使用領域が満杯に近づいていることを知り、不要なファイルを削除することで、圧縮機能付ディスクの空き領域を増やすことができ、これにより物理的な使用領域が満杯になることをできるだけ防ぐことができる。

【0010】圧縮機能付記憶装置は、ホストコンピュータからの要求に応じて、記憶装置の論理的な管理単位に対応している物理的な領域の大きさをホストコンピュータに通知する。ホストコンピュータは、OSの管理情報に基づいて、記憶装置の論理的管理単位とファイルに対応させ、各ファイルがディスク上に占める物理的な領域の大きさを知る。そして、ディスク上に占める物理的な領域の大きい順にファイル名をコンソール出力する。その結果、ユーザはファイルを削除する際に、このコンソール出力を参照して、効率よく空き領域を増やせるファイルを削除対象として指定できる。

【0011】ホストコンピュータは、オペレーティングシステムの管理情報に基づいて、記憶装置上でファイルの割り当てられていない領域を判定し、そこへ圧縮率の良いデータをライトするためのデータライト処理を行なう。ホストコンピュータのデータライト処理は、圧縮機能付記憶装置から要求されたアドレスに圧縮率の良いデータを書き込むよう、圧縮機能付記憶装置にデータライト処理を依頼する。圧縮機能付記憶装置のデータライト処理は、ライト要求のあった圧縮率の良いデータを圧縮して、要求されたアドレスにライトする。そして、要求されたアドレスに元々割り当てられていたディスク上の領域が解放されて、空き領域が確保される。このとき、ライト要求で指定されたアドレスには、元々割り当てられていたディスク上の領域の代わりに、圧縮率の良いデータを圧縮した小さな圧縮データを格納する領域が確保され、多くの場合はディスク上の空き領域が増えることになる。上記の処理は通常のファイル削除と同期して行なうのではなく、ユーザがアプリケーションプログラムを起動することによって実施される。このため、通常の削除処理に余分なオーバーヘッドはかからず、また、OSの改造も必要無い。

【0012】

【実施例】

第1の実施例

5

図1は本発明を適用したコンピュータシステム1000である。まず、構成について説明する。コンピュータシステム1000はデータを圧縮して記憶可能な圧縮機能付ディスク（記憶装置）1030、ホストコンピュータ1020、ホストコンピュータ1020からの情報をユーザーに伝達するためのコンソール1010より構成される。更に、圧縮機能付ディスク1030は、データの圧縮処理や伸長処理を行なう圧縮伸長装置1050、CPU1070、メモリ1060、圧縮データ1140を格納する記憶装置1090より構成される。

【0013】図2は、圧縮機能付ディスク1030において、ホストコンピュータ1020から見える論理ディスク1100と、実際の物理的な記憶装置1090との関係、及び、論理ブロックと物理ブロックの対応付けを示している。

【0014】本実施例では、論理ディスク1100の容量を2GB、記憶装置の容量を圧縮効果を考慮して、1GBとする。論理ディスク1100は、4KBのサイズを持つ論理ブロック1120から構成されており、記憶装置1090は、512Bのサイズを持つ物理ブロック1110から構成されている。ホストコンピュータ1020は、圧縮機能付ディスク1030に対するアクセスを論理ブロック1120単位に行なう。

【0015】次に、論理ディスク1100上の論理ブロックa1121のデータが実際に記憶装置1090に格納される様子を図2に示す。

【0016】論理ブロックa1121のデータである伸長データa1131は圧縮されて圧縮データa1141となり、物理ブロックa1112、物理ブロックb1113、物理ブロックc1114に分割格納される。一般に、圧縮データ1141のサイズは一定ではないため、それを格納するのに必要な物理ブロック1112-4の数は1個～8個となる。ただし、圧縮データ1141が伸長データ1131よりも大きくなることは無いものとする。

【0017】実際に論理ブロック1121と物理ブロック1112-4の対応付けを管理しているテーブルは論理ブロックテーブル1200である。論理ブロックテーブル1200には論理ブロック数分の論理ブロックテーブルエントリ1211-3があり、各論理ブロックテーブルエントリ1211-3には圧縮データサイズ1220、1番目の物理ブロック番号1230～8番目までの物理ブロック番号1290のエントリがある。圧縮データサイズ1220には、その論理ブロック1121の伸長データ1131を圧縮した時の圧縮データ1141のサイズが格納される。また、この圧縮データサイズ1220より、論理ブロック1121に対応する物理ブロック1112-4の数を
40
得ることができる。1番目の物理ブロック番号1230～8番目の物理ブロック番号1290には、圧縮データ1141を格納している最大8個の物理ブロック番号が格納されている。論理ブロック1121に対応する物理ブロック1112-4の数が7個以下の場合には、1番目の物理ブロック番号1230～8番目の物理ブロック番号1290のうち、対応する物理ブロック1112-4が存在しないエントリには、物理ブロック番号としてあり得ない値、たとえば、-1を格納す
50

6

る。また、対応する物理ブロック1112-4が存在しない論理ブロック1121には、圧縮サイズ0を格納し、1番目の物理ブロック番号1230～8番目の物理ブロック番号1290のエントリには値-1を格納することで、その他の場合と区別できるようにする。

【0018】図3は、有効なデータを格納していない空き物理ブロック1112-4を管理する空き物理ブロックテーブルである。空き物理ブロックテーブル1300には、空き物理ブロック数1310と、記憶装置1090上の全ての物理ブロック1112-4に対して、空きかそうでないかをそれぞれ1ビットで示す空き物理ブロックビットマップ1320から構成される。本実施例では、1を空き、0を使用中に対応させる。

【0019】次に、図1を用いて、本発明の概要について述べる。コンピュータシステム1000では、ファイルの削除の際に、ホストコンピュータ1020のファイル削除処理1400が、CPU1070での論理ブロック解放処理1410に対して、削除したファイルに対応する論理ブロック1121の番号を通知し、論理ブロック解放処理1410は、メモリ1060上の論理ブロックテーブル1200と空き物理ブロックテーブル1300とを書き換えて、削除したファイルのデータが保持されないようにする。上記の手順を太い線の矢印で示す。

【0020】太い点線の矢印は、コンピュータシステム1000におけるデータライトの処理を示している。まず、ホストコンピュータ1020のデータライト処理1420が圧縮機能付ディスク1030に対してデータをライトすると、CPU1070のデータライト処理1430は、記憶装置1090にデータを格納できるのであれば、ライトデータを圧縮伸長装置1050で圧縮し、記憶装置1090に書き込んで、メモリ1060上の論理ブロックテーブル1200と空き物理ブロックテーブル1300とを修正し、ホストコンピュータ1020にライト完了を報告する。そして、新たに空きとなった論理ブロック1121があれば、ホスト側のデータライト処理1420は、その論理ブロック1121の番号を圧縮機能付ディスク1030に通知し、ディスク側のデータライト処理1430は、論理ブロックテーブル1200と空き物理ブロックテーブル1300とを修正して、空きとなった論理ブロック1121に対応している物理ブロック1112-4を解放する。一方、記憶装置1090にデータを格納できなければ、ディスク側のデータライト処理1430は、ホストコンピュータ1020にワーニングを報告し、ホスト側のデータライト処理1420はコンソール1010にワーニングを表示する。

【0021】細い破線の矢印は、削除候補のファイル名や容量をコンソールに出力する処理を示している。まず、CPU1070の論理ブロック対応物理ブロック通知処理は、メモリ1060に格納された論理ブロックテーブル1200に基づいて、各論理ブロック1121に対応している物理ブロック数を調べ、ホストコンピュータ1020に通知する。そして、ホストコンピュータ1020の削除候補ファイル選

定処理1440は、OS管理情報1470に基づいて、ファイルと論理ブロック1121-3を対応させ、その対応結果と圧縮機能付きディスク1030から送られてきた情報とを付き合わせて、各ファイルがそれぞれいくつの物理ブロック1112-4に対応しているかを調べる。そして、対応する物理ブロック1112-4数の多い順にファイル名をソートして、コンソール1010に出力する。

【0022】上記の手順を、フローチャートを用いて詳細に説明する。図4、図5を用いて、コンピュータシステム1000におけるファイル削除処理を説明する。図4は、ホストコンピュータ1020におけるファイル削除処理1400のフローチャートである。処理1400は、OSのファイル削除処理に、圧縮機能付きディスクを扱うための機能を追加したものである。まず、ステップ2000では、削除が要求されたファイルが削除される。ステップ2010では、ディスク1030に対して論理ブロック解放コマンドが発行され、削除したファイルが割り当てられていた論理ブロック1121-3の番号が通知される。

【0023】図5は、圧縮機能付きディスクにおける論理ブロック解放処理1410のフローチャートである。まず、ステップ2100では、ホストコンピュータ1020から論理ブロック1121-3の番号を受け取る。ステップ2110では、論理ブロックテーブル1200と空き物理ブロックテーブル1300とを書き換えて、番号を受け取った論理ブロック1120に対応している物理ブロック1112-4を空きにする。

【0024】次に、図6、図7及び図8を用いてコンピュータシステム1000におけるデータライト処理について説明する。図6は、圧縮機能付きディスク1030のデータライト処理1430のフローチャートである。まず、ステップ2200では、ホストコンピュータ1020よりライトデータを受けとって、メモリ1060に格納する。このライトデータは非圧縮の状態であるため、伸長データ1131と呼ぶ。次に、ステップ2210では、圧縮伸長装置1050は、メモリ1060上の伸長データ1131を圧縮して圧縮データ1141を生成し、メモリ1060に格納する。そして、CPU1070は、空き物理ブロックビットマップ1320を用いて、圧縮データ1141を格納するのに必要な数の空き物理ブロック1110を探す(ステップ2220)。メモリ1060上の圧縮データ1140が、ステップ2220で見つけた空き物理ブロック1110にライトされる(ステップ2230)。

【0025】ステップ2240では、CPU1070はライト要求のあった論理ブロック1121に対応する論理ブロックテーブルエントリ1211-3を書き換え、論理ブロック1121と物理ブロック1112-4を対応させる。さらに、ステップ2250で、CPU1070は、空き物理ブロックテーブル1300に格納された空き物理ブロック数1310からライトした物理ブロック数を引き、空き物理ブロックビットマップ1320の中のライト済の物理ブロックに対応するビットに使用中を示す0を設定する。また、ライトが要求された論理ブロック1121に元々対応していた物理ブロック1112-4は空き

として解放され、空き物理ブロック数1310にその数が加算され、空き物理ブロックビットマップ1320の対応するビットに1が設定される。

【0026】ステップ2251では、CPU1070が、空き物理ブロックテーブル1300の空き物理ブロック数1310に基づいて、ライトデータを格納できる容量の空き物理ブロック1112-4があるかどうかを判定する。そして、ブロック1112-4が不足していればステップ2252にジャンプする。ブロック1112-4が不足していなければ、ステップ2260では、CPU1070は、空き物理ブロックテーブル1300の空き物理ブロック数1310に基づいて、空き物理ブロック1110が十分に残っているかどうかを判定する。本実施例では、空き物理ブロック1110数が全物理ブロック1110数の10%以上あれば十分に残っていると判断するものとする。

【0027】そして、空き物理ブロック1110の数が十分に残っていると判定されればステップ2280にジャンプする。空き物理ブロック1110の数が十分に残っていないと判定された場合、ステップ2270で、CPU1070はホストコンピュータ1020にライト完了報告を行なうが、このとき、空き物理ブロック1110が少ないことを示すワーニングを報告する。ステップ2290では、ホストコンピュータ1020から論理ブロック解放要求が来ているかどうかを判定される。論理ブロック解放要求が来ていなければ、圧縮機能付きディスク1030におけるライト処理は終了する。論理ブロック解放要求が来ていれば、ステップ2300では、図5に示した論理ブロック解放処理1410が行なわれ、圧縮機能付きディスク1030におけるライト処理は終了する。

【0028】一方、ステップ2280では、CPU1070は、ホストコンピュータ1020にライト完了報告を行なうが、このとき、ワーニング報告は行なわず、正常終了を報告してステップ2290にジャンプする。

【0029】他方、ステップ2252では、ホストコンピュータ1020に空き物理ブロック1110の数が不足していて、ライトデータをディスクに格納できないことを報告し、圧縮機能付きディスク1030におけるデータライト処理は終了する。

【0030】図7は、ホストコンピュータ1020のデータライト処理1420のフローチャートである。まず、ステップ2400では、圧縮機能付きディスク1030に対してライト要求が発行される。ステップ2410で、圧縮機能付きディスク1030にライトデータが送られる。ステップ2420では、圧縮機能付きディスク1030のライト処理の完了待ちの状態となる。ステップ2430で、圧縮機能付きディスク1030からの完了報告が受け取られる。ステップ2440で、圧縮機能付きディスク1030から、空き物理ブロック1110が少ないことを示すワーニングが報告されたかどうかを判定され、ワーニングがなければ、ステップ2460にジャンプする。ワーニングが報告されている場合、ステップ2450では、コ

9

ンソール1010にワーニングが出力され、ユーザに対処方法を示唆する。ステップ2460では、新たに空きとなった論理ブロックが存在するかどうか判定される。上記の処理は、ファイルの上書き等により、OSがファイルに割り当てる論理ブロック1120を変更した場合に生ずる。新たに空きとなった論理ブロック1120が存在しなければ、ホストコンピュータ1020のデータライト処理は終了する。空き論理ブロック1120が存在すれば、ステップ2470では、圧縮機能付ディスク1030に論理ブロック1120の解放要求が発行され、空きとなった論理ブロック1120の番号が送信されて、ホストコンピュータ1020のデータライト処理は終了する。

【0031】以上で述べたように、ホストコンピュータ1020でファイルを削除する度に、あるいは、データライト処理によって、ファイルに割り当てられている論理ブロック1120が変更される度に、圧縮機能付ディスク1030に、データ保持の必要が無くなった論理ブロック1120を通知して、不要なデータが保持されないようにすることで、圧縮機能付ディスク1030の利用効率を上げることができる。

【0032】図8はコンソール1010に表示するワーニングの例である。「ディスクの空き領域が少なくなっています。」の表示によって、空き物理ブロック1110が少ないことをユーザに通知している。「不要なファイルを削除して下さい。」の表示では、ユーザに今後の対処方法を示唆している。ユーザーは不要と判断したファイルを削除することで、物理ブロック内の空き物理ブロック1110数を増やすことができる。

【0033】

「使用中物理容量: XXXXKB

使用可能物理容量: YYYYKB

トータル物理容量: ZZZZKB」

の表示からユーザは、記憶装置1090の利用状況を知ることができる。この表示により、ユーザはファイルの削減量の目安を知ることができる。

【0034】そして、ユーザは不要なファイルを削除し、それでも使用可能物理容量が十分でない場合には、ファイルをいくつかを他のディスクやテープ等にコピーして圧縮機能付ディスク1030から削除して、空き物理ブロック1110を増やすことで、物理的な使用領域の満杯を防ぐことができる。

【0035】最後に、図9及び図13を用いて、ユーザに削除候補のファイルを示す処理を説明する。ユーザが削除するファイルをいくつか選択する際に、圧縮率の良いファイルばかりが選択されたときと、圧縮率の悪いファイルばかりが選択されたときとは、新たに使用可能となる空き物理ブロック1110の数に大きな違いが出てしまう。従って、本実施例では、ファイルが消去されたときに、できるだけ多くの物理ブロック1110が空きとなるようなファイルをユーザが削除対象として選択できるよ

10

うに、コンソール1010に、対応する物理ブロック数の多い順にファイル名が出力される。

【0036】図9は、圧縮機能付ディスク1030の論理ブロック対応物理ブロック数通知処理1450のフローチャートである。まず、ステップ2600では、論理ブロックテーブル1200に基づいて、各論理ブロック1120に対応している物理ブロック1110の数を調べる。ステップ2610では、各論理ブロック1120に対応する物理ブロック1110の数がホストコンピュータ1020に送られる。

【0037】図13は、ホストコンピュータの削除候補ファイル選定処理1440のフローチャートである。まず、ステップ2700では、圧縮機能付ディスク1030に、各論理ブロック1120に対応している物理ブロック1110の数を通知するような要求が発行される。ステップ2710では、圧縮機能付ディスク1030より、各論理ブロック1120に対応している物理ブロック1110数のデータが受け取られる。ステップ2720では、図10に示すように、OS管理情報1470に基づいて、論理ブロック1120とファイルとを関連付けて、各ファイルに対応する物理ブロック1110の数が調べられる。例えば、ホストコンピュータ1020のOSがMS-DOSである場合には、FAT (File Allocation Table) に基づいて、ファイルと論理ブロック1120との対応関係を知ることができる。最後に、ステップ2730では、対応する物理ブロック1110数の多い順番にファイル名がコンソール1010に出力される。

【0038】図11は消去候補ファイルの画面出力例である。各ファイル毎に、ファイル名、圧縮後サイズ、及び圧縮前サイズからなる情報が、圧縮後ファイルサイズの順番に表示される。ファイルの圧縮後のサイズは、対応する物理ブロック1110数に物理ブロック1110サイズを掛ければ得られる。また、ファイルの圧縮前のサイズは、OS管理情報1470から得られる。ユーザは、コンソールを見て、どのファイルが多く圧縮機能付ディスク1030を占有しているのかを知ることができ、これを選択材料の一つとして、削除すべきファイルを選択する。

【0039】第2の実施例

第1の実施例では、圧縮機能付ディスクの利用効率の問題を解決するために、ファイルを削除する度に、削除の指示をディスクに通知し、削除したファイルに割り当てられていた論理ブロック1120に対応する物理ブロック1110を解放していた。しかし、上記の方法では、ファイルの削除処理やライト処理と同期して実行されるため、処理の同期に伴うオーバーヘッドが大きくなるという欠点があった。また、データを保持する必要の無くなった論理ブロック1120をディスクに通知し、対応する物理ブロック1110を空きとするための処理をOSに作り込む必要があり、その改造規模が大きかった。

【0040】図12は本実施例に対応する発明の概要である。本実施例では、ファイルの消去処理やライト処理のオーバーヘッドを減らし、OSの改造規模を小さくす

るため、ファイルの削除の度にそれをディスクに通知するのではなく、使用可能な空き物理ブロック1110が少なくなってきたときに、データ保持の必要が無い全ての論理ブロック1120にまとめて圧縮率の良いデータを書き込むことで、空き物理ブロック1110数を増やす空き領域回復アプリケーション1460を設ける。処理1460により、使用可能な空き物理ブロック1110数を増やし、圧縮機能付ディスク1030の利用効率の問題に対処する。

【0041】太い点線の矢印は、コンピュータシステム1000のデータライト処理を示す。まず、ホストコンピュータ1020のデータライト処理A1421が圧縮機能付ディスク1030にデータをライトすると、CPU1070のデータライト処理A1431は、記憶装置1090上にデータを格納できれば、ライトデータを圧縮伸長装置1050で圧縮し、その結果を記憶装置1090に書き込んで、メモリ1060の論理ブロックテーブル1200と空き物理ブロックテーブル1300を修正し、ホストコンピュータ1020にライト完了を報告する。一方、記憶装置1090にデータ格納できなければ、ディスク側データライト処理A1431はホストコンピュータ1020にワーニングを報告し、ホスト側データライト処理A1421はコンソール1010にワーニングを表示する。

【0042】細い破線の矢印は、削除候補のファイルをコンソール出力する処理を示す。まず、CPU1070の論理ブロックに対応する物理ブロックを通知する処理は、メモリ1060にある論理ブロックテーブル1200に基づいて、各論理ブロック1120に対応する物理ブロック数を調べ、ホストコンピュータ1020に通知する。ホストコンピュータ1020の削除候補ファイル選定処理1440はOS管理情報1470に基づいて、ファイルと論理ブロック1120とを対応させ、これと送られてきた情報とを付き合わせて、各ファイルがそれぞれいくつの物理ブロック1110に対応しているかを調べる。対応する物理ブロック1110数の多い順に、ファイル名がソートされて、コンソール1010に出力される。

【0043】細い実線の矢印は、ホストコンピュータ1020の空き領域回復アプリケーション1460の処理を示す。まず、OS管理情報1470に基づいて、OSが使用していない論理ブロック1120が判定され、全ての未使用論理ブロック1120にオール0のデータを書き込むよう、ホスト側データライト処理A1421にライト要求が出される。

【0044】図14は、ファイルを削除され、格納データが不要となった論理ブロック1120へ、圧縮率の良いデータを書き込む前後で、圧縮機能付ディスク1030の空き物理ブロック1110が増える様子の具体例を示す。

【0045】ファイルの消去前、論理ブロックk1125には伸長データm1135が格納されている。伸長データm1135は、圧縮されて圧縮データm1145となり、6個の物理ブロック1110に格納される。ファイル削除後に、圧縮率の良い伸長データn1136が書き込まれると、伸長データn1136は圧縮されて圧縮データn1146となり、圧縮データn11

46は一つの物理ブロック1110に格納される。このとき、残りの5個の物理ブロック1110は空き物理ブロック1110となる。

【0046】以下、第1の実施例と異なる点を説明する。本実施例では、ファイルの削除の度に、削除されたファイルに対応する論理ブロック1120をディスクに通知しないので、図5に示した論理ブロック解放処理1410と、図4のステップ2010は必要無いので、図15のようになる。

【0047】また、ファイルライト時にも、データ保持の必要がなくなった論理ブロック1120をディスクに通知しないので、図6のステップ2290とステップ2300、及び、図7のステップ2460とステップ2470は必要無い。図6及び図7はそれぞれ、図16、図17のようになる。

【0048】本実施例に対応した、コンソール出力するワーニング画面は図18のようになる。「空き領域回復アプリケーションを走らせて下さい」の表示が、以下の理由により追加される。本実施例では、ファイルを削除しただけでは、圧縮機能付ディスク内の使用可能な物理ブロック1110を増加させることはできないため、空き領域回復アプリケーション1460を実行させて、削除したファイルのデータが入っている物理ブロック1110を解放してやる必要があるためである。

【0049】図19は空き領域回復アプリケーション1460のフローチャートである。ステップ2500では、ホストコンピュータ1020のOSの管理領域に基づいて、使用していない論理ブロック1120が判定される。例えば、ホストコンピュータ1020のOSがMS-DOSであれば、FAT (File Allocation Table) に基づいて、上記の判断が可能である。他のOSの場合もファイルや空きブロックを管理する管理情報をOSが持っている場合には、この管理情報を参照して、使用していない論理ブロック1120を判定できる。ステップ2510では、未使用と判断された論理ブロック1120に、オール0のデータが書き込まれる。ここで、オール0のデータとは、圧縮率の良いデータの一例として示しており、比較的圧縮率の良いデータであればどのようなデータでもかまわない。ただし、圧縮データ1140が1つの物理ブロック1110に納まるデータが望ましい。本実施例では、オール0のデータを圧縮することで、一つの物理ブロックに納まるサイズになるものとする。

【0050】上記のように、空き領域回復アプリケーション1460を実行させることにより、圧縮機能付ディスク1030において、削除されたファイルのデータを持続している物理ブロック1110を解放し、空き物理ブロック1110を増やすことができる。

【0051】

【発明の効果】本発明によれば、圧縮機能付記憶装置は、対応する論理ブロックがなく、かつ、新たな論理ブロックの割り当てが可能な物理ブロック数が少なくなっ

て、物理的な使用領域が満杯に近づいたと判定されたとき、ホストコンピュータにワーニングを報告し、ホストコンピュータはそれをコンソールに出力する。その結果、ユーザはコンソールに出力されたワーニングを見て、物理的な使用領域が満杯に近づいていることを知り、不要なファイルを削除することで、圧縮機能付ディスクの空き領域を増やすことができ、これにより物理的な使用領域が満杯になることをできるだけ防ぐことができる。

【0052】圧縮機能付記憶装置は、ホストコンピュータからの要求に応じて、記憶装置の論理的な管理単位に対応している物理的な領域の大きさをホストコンピュータに通知する。ホストコンピュータは、OSの管理情報に基づいて、記憶装置の論理的管理単位とファイルに対応させ、各ファイルがディスク上に占める物理的な領域の大きさを知る。そして、ディスク上に占める物理的な領域の大きい順にファイル名をコンソール出力する。その結果、ユーザはファイルを削除する際に、このコンソール出力を参照して、効率よく空き領域を増やせるファイルを削除対象として指定できる。

【0053】ホストコンピュータは、オペレーティングシステムの管理情報に基づいて、記憶装置上でファイルの割り当てられていない領域を判定し、そこへ圧縮率の良いデータをライトするためのデータライト処理を行なう。ホストコンピュータのデータライト処理は、圧縮機能付記憶装置から要求されたアドレスに圧縮率の良いデータを書き込むよう、圧縮機能付記憶装置にデータライト処理を依頼する。圧縮機能付記憶装置のデータライト処理は、ライト要求のあった圧縮率の良いデータを圧縮して、要求されたアドレスにライトする。そして、要求されたアドレスに元々割り当てられていたディスク上の領域が解放されて、空き領域が確保される。このとき、ライト要求で指定されたアドレスには、元々割り当てられていたディスク上の領域の代わりに、圧縮率の良いデータを圧縮した小さな圧縮データを格納する領域が確保され、多くの場合はディスク上の空き領域が増えることになる。上記の処理は通常のファイル削除と同期して行なうのではなく、ユーザがアプリケーションプログラムを起動することによって実施される。このため、通常の削除処理に余分なオーバーヘッドはかからず、また、OS

の改造も必要無い。

【図面の簡単な説明】

【図1】第1の実施例のコンピュータシステムの構成例と発明の概要を示す。

【図2】論理ブロックと物理ブロックの対応関係を示す。

【図3】空き物理ブロックテーブルを示す。

【図4】ホストコンピュータのファイル削除処理のフローチャートを示す。

【図5】圧縮機能付ディスクの論理ブロック解放処理のフローチャートを示す。

【図6】圧縮機能付ディスクのディスク側ライト処理のフローチャートを示す。

【図7】ホストコンピュータにおけるホスト側ライト処理のフローチャートを示す。

【図8】本実施例におけるコンソールへのワーニング出力例を示す。

【図9】圧縮機能付ディスクにおける、論理ブロックに対応する物理ブロック数通知処理のフローチャートを示す。

【図10】ホストコンピュータの削除候補ファイル選定処理のフローチャートを示す。

【図11】削除候補ファイル選定処理がOS管理情報を調べる様子を示す。

【図12】消去候補ファイルのコンソール出力例を示す。

【図13】第2の実施例のコンピュータシステムの構成例と発明の概要を示す。

【図14】削除されたファイルに圧縮率の良いデータをライトする前後の空き物理ブロックの増加を示す。

【図15】ホストコンピュータのファイル削除処理Aのフローチャートを示す。

【図16】圧縮機能付ディスクにおけるディスク側ライト処理のフローチャートを示す。

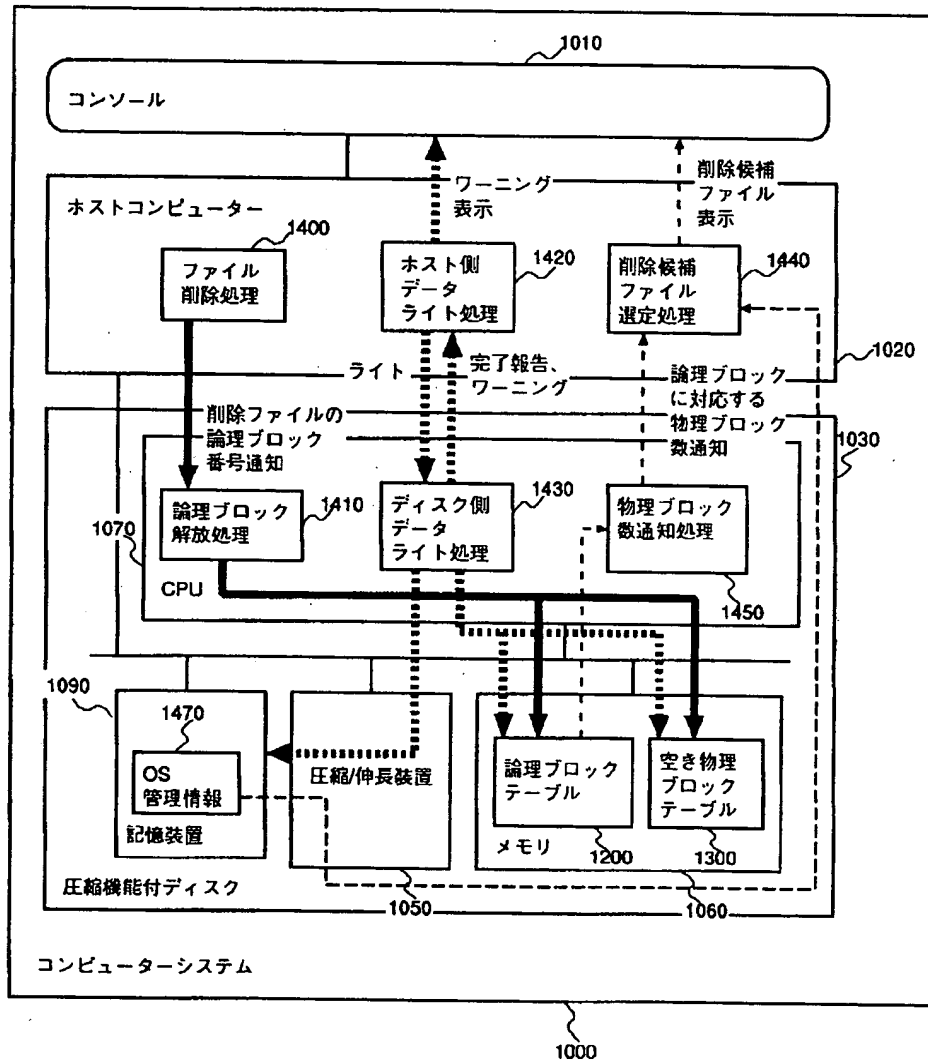
【図17】ホストコンピュータにおけるホスト側ライト処理Aのフローチャートを示す。

【図18】第2の実施例におけるコンソールへのワーニング出力例を示す。

【図19】満杯対策のアプリケーションのフローチャートを示す。

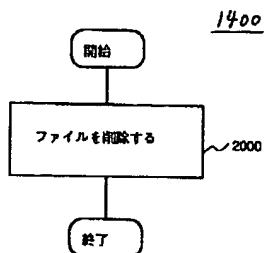
【図1】

図1



【図15】

図15



【図 3】

3

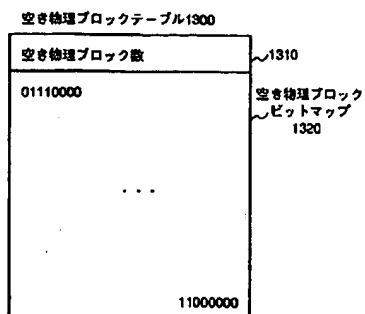
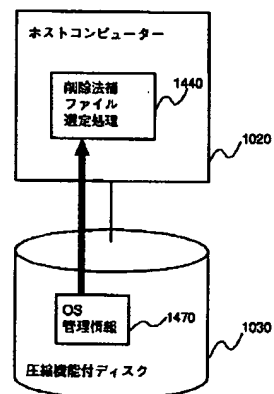
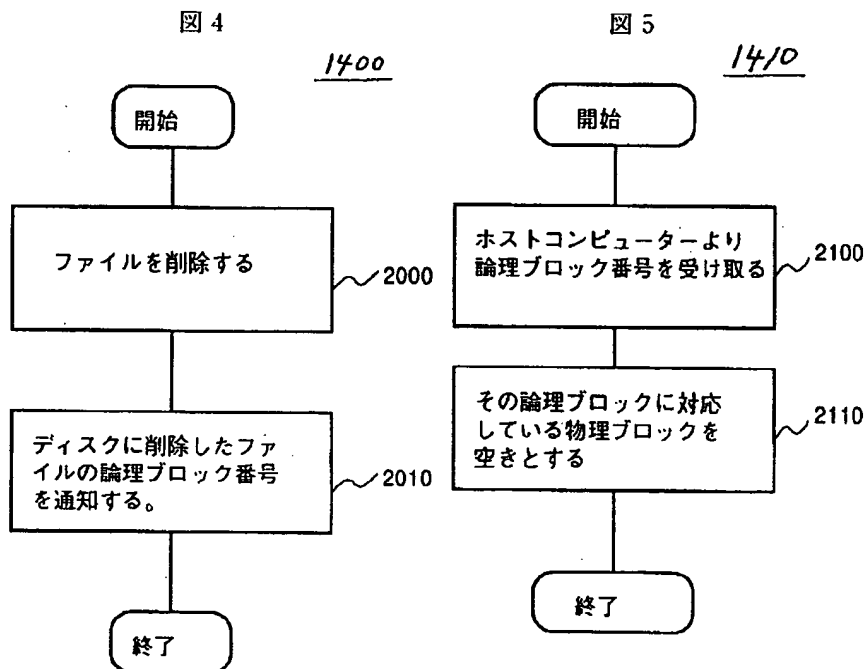


图 10

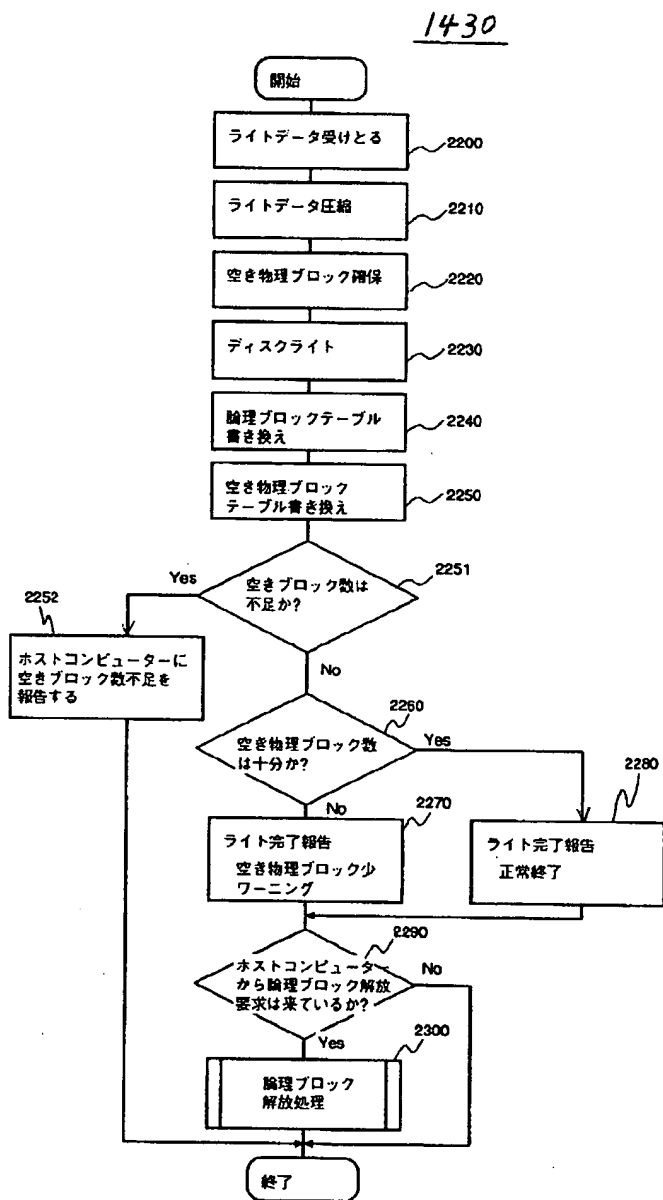


【图5】



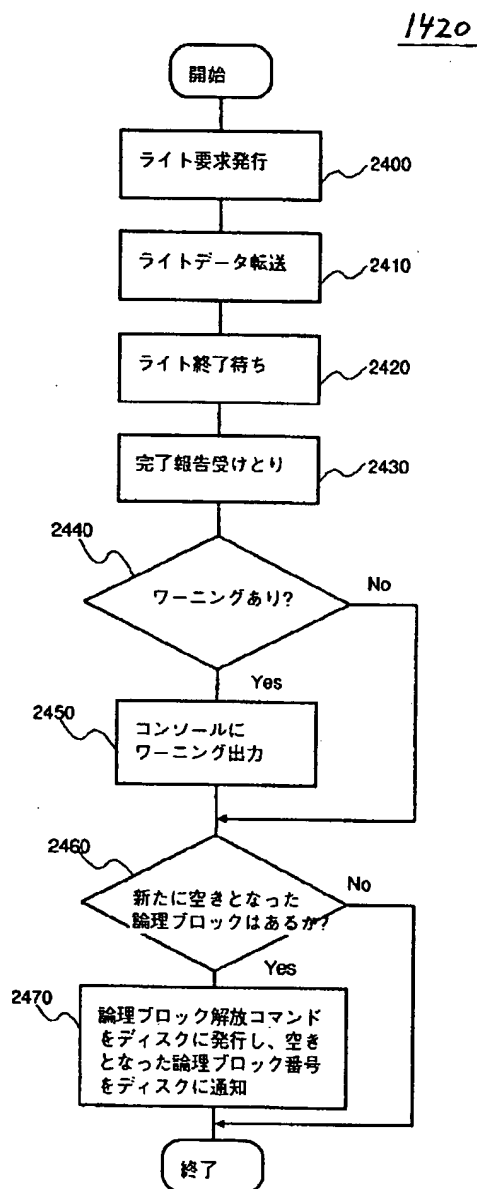
【図6】

図 6



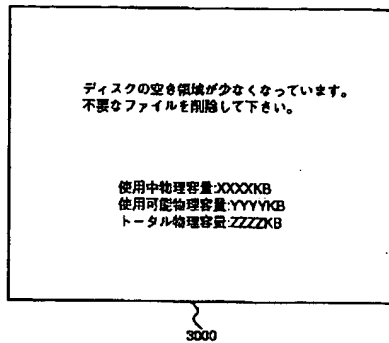
【図7】

図 7



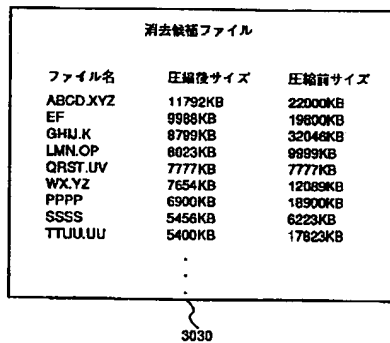
【図8】

図 8



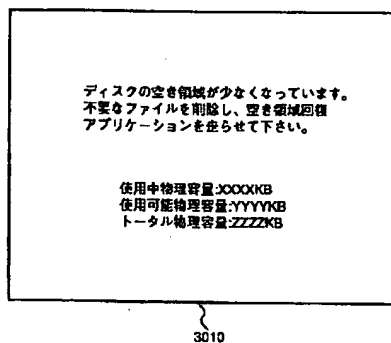
【図11】

図 1 1



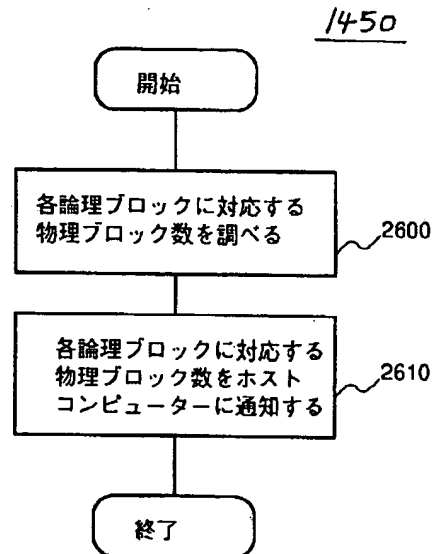
【図18】

図 1 8



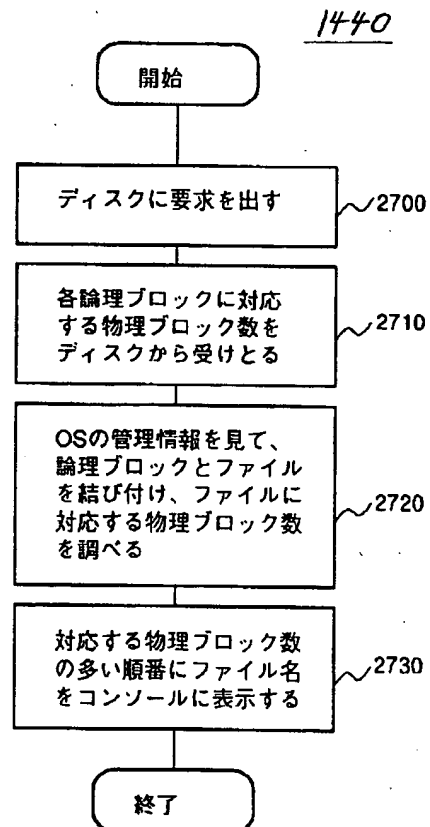
【図9】

図 9



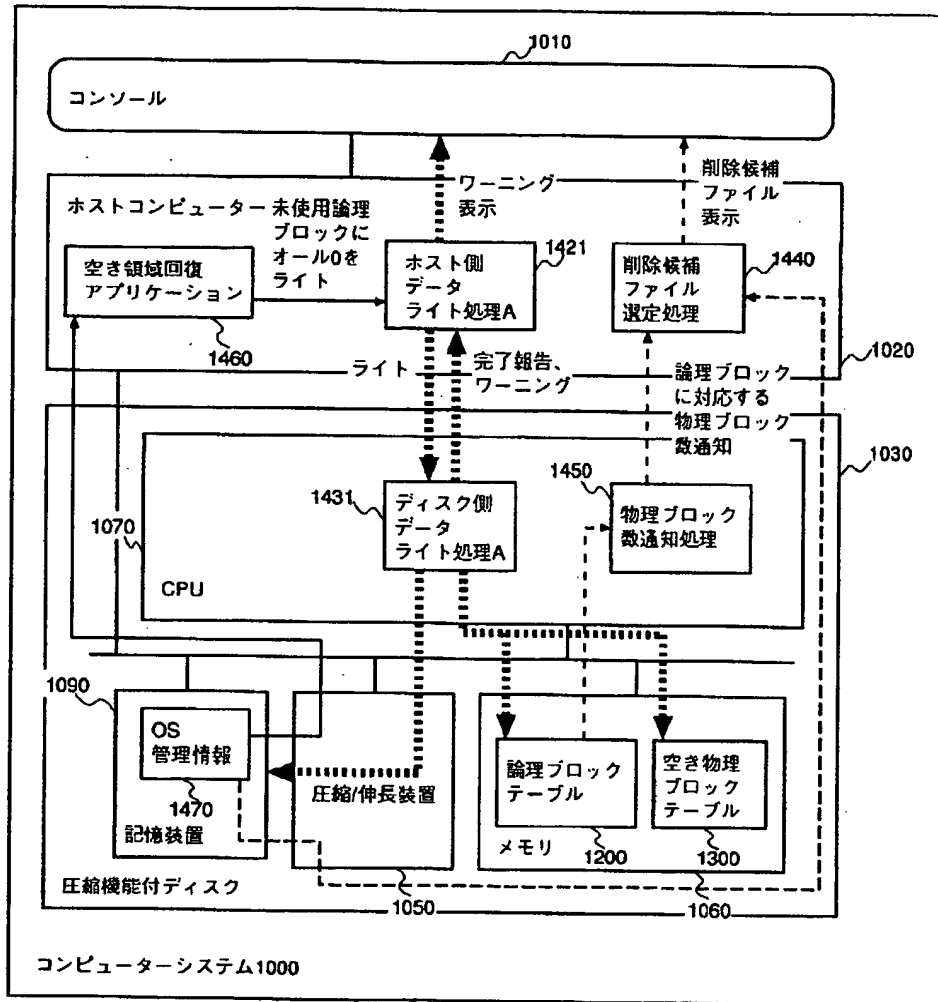
【図13】

図 1 3



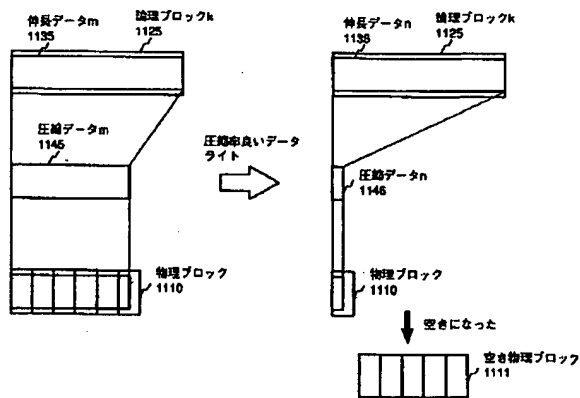
【図12】

図12



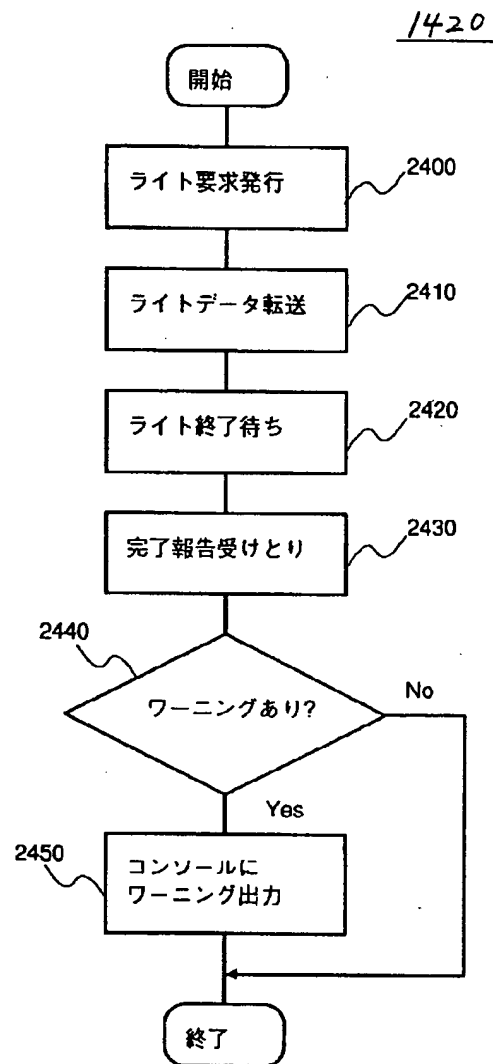
【図14】

図14



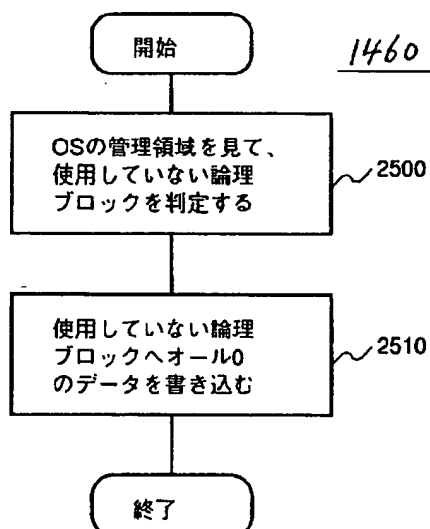
【図17】

図17



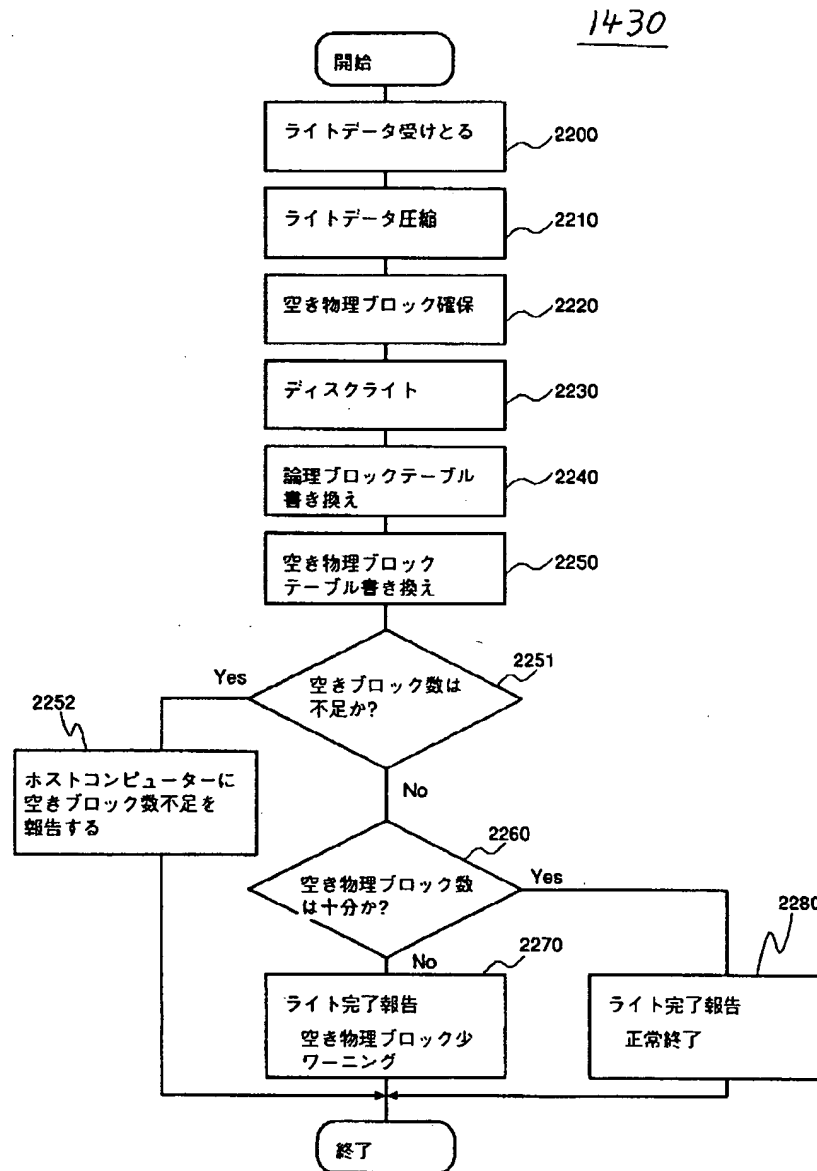
【図19】

図19



【図16】

図16



フロントページの続き

(72)発明者 山形 博健
 神奈川県小田原市国府津2880番地 株式会
 社日立製作所ストレージシステム事業部内

(72)発明者 小橋 徹三
 神奈川県小田原市国府津2880番地 株式会
 社日立製作所ストレージシステム事業部内

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☒ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.